PixGS: Pixel-Space Diffusion for Direct 3D Gaussian Splat Generation

Cao Duy Phong Nguyen

Qualcomm AI Research*
{duycao, phongnh}@qti.qualcomm.com

Abstract. Recent advances in 3D content generation from text or images have achieved impressive results, yet view inconsistency from 2D generators and the scarcity of high-quality 3D data remain significant bottlenecks. Existing solutions [21, 28] typically adapt large-scale pretrained text-to-image latent diffusion models to generate 3D Gaussian Splats (3DGS). However, these approaches often rely on training complex cascade pipelines that are computationally expensive and scalability-limited. Most critically, the quality of generated 3D assets is inherently constrained by each component capacity and compressed latent space, leading to decoding artifacts and accumulated errors. To address these limitations, we propose **PixGS**, a single-stage pipeline for direct high-quality 3DGS generation, which leverages recent advances in pixel-space diffusion to bypass lossy latent compression while still benefiting from the vast 2D generative priors. By directly denoising 3D Gaussian attributes at each timestep, our method enables precise, splat-level regularization of both appearance and geometry. Furthermore, we introduce a comprehensive supervision strategy that incorporates surface normals, depth, and high-frequency structural information, which is often overlooked in prior works. Experiments demonstrate that PixGS outperforms current state-of-the-art methods while maintaining a fast inference speed ($\approx 1s$ on a single A100 GPU), offering a robust and efficient alternative to multi-stage generation pipelines.

Keywords: 3D Generation · Gaussian Splatting · Generative Model

1 Introduction

While recent 3D generative advances often favor mesh-based representations [12, 18, 43, 58] for their geometric clarity in animation and printing, domains such as VR/AR, gaming and film prioritize visual realism and rendering efficiency. This has cemented 3D Gaussian Splats (3DGS) [16, 17] as a pivotal representation, fueling research efforts to integrate its superior rendering capabilities into robust generative pipelines.

Several existing methods [40, 46, 49] attempt to synthesize 3DGS using multi-view-aware 2D diffusion models [23, 36] to generate consistent images of an object, followed by a separate 3D reconstruction stage. While promising, the final

* Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

3D quality is strictly capped by the upstream 2D generator, and any slight inconsistency in the generated views leads to artifacts (floaters) during reconstruction.

A recent promising direction involves directly generating 3DGS in an image-like format. Inspired by generalizable 3DGS reconstruction techniques [3, 39], DiffSplat [21] finetunes latent diffusion models to generate 2D tensors where each pixel encodes the attributes of a 3D Gaussian. While this provides multi-view consistency and leverages 2D priors, it introduces a significant architectural bottleneck. The pipeline requires training three interdependent models: a 3DGS reconstructor, a VAE to compress attributes into a latent space, and a latent diffusion model. This cascaded training regime is computationally expensive and introduces substantial engineering overhead, as each stage must converge reasonably well before the next can proceed, making the overall system fragile and difficult to scale. Also, to mitigate the computation costs of repeated Gaussian decoding during diffusion training, DiffSplat employs an additional lightweight decoder, trained from scratch, which reduces memory usage at the expense of increased system complexity. Moreover, reliance on a compressed latent space inherently limits the representational capacity and fine-grained detail of the generated 3DGS.

We propose PixGS, a single-stage pipeline for high-quality 3DGS generation, designed to address the limitations of previous works. PixGS leverages advances in large-scale Pixel-space Image Diffusion [5, 25, 26, 44, 55] models trained on extensive image corpora. To capitalize on the powerful 3D geometry estimation priors inherent in web-scale image diffusion models demonstrated in prior research [10, 15, 19, 21], we represent 3D objects using Gaussian attribute tensors, which are 2D grids where each pixel encodes the geometric and textural properties of the 3DGS. Instead of relying on 3D latent space, our generative model learns and operates directly at the splat distribution, denoising and regularizing Gaussians at each timestep. To leverage this, we introduce a comprehensive loss set, including depth and normal supervision, as well as Laplacian of Gaussian [27] (LoG) loss for high-frequency feature regularization. By bypassing the training requirements of VAE latent compression and integrating LoG loss, we successfully capture intricate details that previous methods often fail to recover (such as text and thin structures) while maintaining computational efficiency. Alongside the applied loss functions, we introduce a three-phase training schedule that decouples the generative model from the constraints of the reconstruction model. Our contributions can be summarized as follows:

- We introduce PixGS, a single-stage pipeline for high-quality 3DGS generation that bypasses the limitations of latent compression and is highly scalable.
- We adapt 2D pixel-space diffusion to operate on Gaussian attribute tensors while preserving powerful pre-trained 2D priors to ensure rapid convergence.
- We integrate a suite of loss functions designed for the direct supervision of 3DGS geometry and appearance, focusing on high-frequency details and a three-phase training scheme that progressively enhances generation quality beyond the capabilities of the initial 3DGS reconstructor.

– Extensive experiments demonstrating the strong performance of PixGS, supported by ablation studies that validate the effectiveness of each design choice.

2 Related Work

Optimization-based generative models. Early breakthroughs in 3D generation were dominated by optimization-based methods [20,33]. These approaches leverage the visual richness of 2D diffusion models to iteratively refine 3D representations, typically NeRFs, through Score Distillation Sampling (SDS), ensuring their rendered views align with a text prompt. Following the introduction of 3D Gaussian Splatting, frameworks [6,41,53] adapted these optimization strategies to 3DGS representation, significantly accelerating the synthesis process. While these methods require no ground-truth 3D data and inherit the creative capacity of 2D foundations, they are notoriously slow due to their per-scene optimization requirement. Furthermore, they frequently suffer from over-smoothed geometry or multi-view inconsistencies, such as “Janus" problem [45].

3D-native generative models To facilitate feed-forward synthesis, 3D-native generative models [14, 29, 57, 60] have emerged to learn 3D shape distributions directly from geometric data. These approaches achieve rapid inference and maintain superior geometric consistency compared to optimization-based methods. However, they are fundamentally constrained by the scale of available 3D datasets [7, 8], which are orders of magnitude smaller than web-scale image corpora. Consequently, capturing high-quality textures and complex topologies often necessitates massive architectural capacity, as seen in recent frameworks [43,47,48,58]. Despite their scale, these models often struggle to generalize beyond their specific training distributions compared to methods that leverage 2D foundation priors.

Reconstruction-based 3D Generative Models A prominent direction involves 2D prior utilization through a multi-stage process: first producing multi-view-consistent 2D images [23,36,42] and then mapping them to 3D space via a separate reconstructor [4,40,46,49,51]. However, this sequential architecture introduces a significant geometric bottleneck. Because the reconstructor treats the generated images as fixed ground truth, any subtle inconsistencies in perspective or texture between views lead to accumulated errors and “floaters" highlighting the inherent difficulty of maintaining geometric integrity in decoupled pipelines.

Structured 3DGS Generation Building on generalizable reconstruction frameworks [3,39,54], recent methods [21,28] attempt to directly synthesize 3DGS as structured 2D attribute tensors. While this image-like format facilitates multi-view consistency, it typically necessitates a complex, multi-stage pipeline. The cascaded regime introduces substantial engineering overhead and training fragility, as it requires the sequential convergence of interdependent components. Conversely, single-stage approaches like DiffusionGS [2] attempt to generate 3DGS directly through a diffusion process. However, these models typically rely on specialized architectures trained entirely from scratch on synthetic 3D datasets. By

discarding pre-trained 2D priors, these models suffer from slow convergence and limited generalization, as they must learn basic visual semantics and appearance cues solely from a relatively small 3D training corpus. Our method addresses the limitations of previous works by introducing a single-stage framework leveraging large-scale 2D priors, with fast inference that produces high-quality 3D content.

3 Background

3.1 Pixel-Space Diffusion

Pixel-space diffusion models [5, 25, 26, 44, 55] generate images by learning a velocity field $v_\theta(\mathbf{x}_t, t)$ on the raw pixel manifold $\mathbf{x} \in \mathbb{R}^{H\times W\times 3}$. Following the Flow Matching formulation [22], the forward process defines a probability path $\mathbf{x}_t = t\mathbf{x}_1 + (1-t)\mathbf{x}_0$ between data $\mathbf{x}_1$ and noise $\mathbf{x}_0 \sim \mathcal{N}(0, \mathbf{I})$, with $t \in [0, 1]$, where the model is trained to predict the constant velocity $\mathbf{v}_t = \mathbf{x}_1 - \mathbf{x}_0$. To maintain computational efficiency without a VAE, these models partition the image into large patches $\mathbf{X} = \{\mathbf{X}^n\}_{n=1}^N$ (e.g., 16×16).

However, standard linear decoders struggle to recover high-frequency details from large, compressed tokens. PixNerd [44] improves this by introducing a patch-wise neural field. For each patch token $\mathbf{X}^n$, the transformer predicts the weights $\mathcal{W}^n$ of a local MLP: $\mathcal{W}^n = \text{Linear}(\text{SiLU}(\mathbf{X}^n))$. The velocity $\mathbf{v}^n(i, j)$ for a specific pixel coordinate (i, j) is then decoded as:

$$\mathbf{v}^n(i,j) = \text{MLP}\left(\text{Concat}[\text{PE}(i,j), \mathbf{x}_t^n(i,j)] \mid \mathcal{W}^n\right), \tag{1}$$

where PE is a coordinate encoding (DCT-basis [1]). By replacing fixed linear projections with coordinate-conditioned neural fields, PixNerd achieves high-fidelity, single-stage generation directly in pixel space.

3.2 Splatter Image

Splatter Image [39] represents a 3D scene as an image-structured collection of 3D Gaussians. Unlike unordered point clouds, it utilizes a 2D image of size $H \times W$ as a container, where each pixel u stores the parameters of exactly one 3D Gaussian, where dimension K include the Gaussian properties: opacity α, color c, covariance Σ, and center μ. Subsequently, an image-to-image network is employed to map the input image into a gaussian tensor $M \in \mathbb{R}^{K\times H\times W}$.

To reconstruct the 3D geometry from a single view, the center μ of a Gaussian at pixel $u = (u_1, u_2)$ is parameterized by a predicted depth d and a 3D offset $\Delta = (\Delta_x, \Delta_y, \Delta_z)$:

$$\mu = \begin{bmatrix} u_1 d + \Delta_x \\ u_2 d + \Delta_y \\ d + \Delta_z \end{bmatrix}. \tag{2}$$

While the Gaussians are initially aligned with camera rays, the predicted 3D offsets Δ allow the network to "splat" Gaussians into unobserved regions, enabling a full 360° reconstruction from a single viewpoint.

3.3 3DGS Rendering

3D Gaussian Splatting (3DGS) [16, 17] represents a 3D scene using a set of explicit, translucent Gaussian primitives. Each Gaussian is parameterized by its center $\mu \in \mathbb{R}^3$, opacity $\alpha \in [0, 1]$, color c (often represented via Spherical Harmonics), and a 3D covariance matrix Σ. To ensure physical validity during optimization, Σ is factored into a scaling matrix S and a rotation matrix R as $\Sigma = RSS^\top R^\top$.

To render the scene from a given viewpoint, 3D Gaussians are projected into 2D image space. Using a local affine approximation of the camera transformation, the projected 2D covariance Σ' is computed as:

$$\Sigma' = JW\Sigma W^\top J^\top, \tag{3}$$

where W is the world-to-camera transformation matrix and J is the Jacobian of the projective transformation. The final color C for a pixel is determined by sorting the N overlapping Gaussians by depth and performing differentiable α-blending:

$$C = \sum_{i=1}^{N} c_i \sigma_i \prod_{j=1}^{i-1} (1 - \sigma_j), \tag{4}$$

where σ_i is the density of the i-th Gaussian at that pixel, calculated by evaluating the 2D Gaussian distribution multiplied by the opacity α_i.

4 Methodology

PixGS is a single-stage, pixel-space diffusion model that directly generates 3DGS parameters bypassing the limitations of a latent bottleneck. Our motivation for adopting an image-like 3D representation (Sec. 4.1) is to leverage established 2D image priors while avoiding the information loss associated with VAE compression. We introduce a suite of tailored design choices (Sec. 4.2), objectives (Sec. 4.3), and training schemes (Sec. 4.4) that adapt pixel-level diffusion to the requirements of 3DGS generation.

4.1 3D Representation

Following DiffSplat [21], we parameterize our 3D representation as a grid of pixel-aligned Gaussians, where each primitive $\mathbf{g}_i \in \mathbb{R}^{12}$ consists of color $\mathbf{c}$, scale $\mathbf{s}$, rotation $\mathbf{r}$, opacity o, and depth d. The 3D position $\mathbf{x}$ can be unprojected from the 2D image plane using the camera's intrinsic $\mathbf{K}$ and extrinsic $\{\mathbf{R}, \mathbf{t}\}$ matrices: $\mathbf{x} = \mathbf{R}^\top \mathbf{K}^{-1}[\mathbf{u}|d] - \mathbf{t}$, with $[\mathbf{u}|d]$ is homogeneous pixel coordinates $\mathbf{u} \in \mathbb{R}^2$ with depth.

We represent each 3D object as $\mathcal{G} \in \mathbb{R}^{V_{in} \times g \times H \times W}$ tensor, where $H \times W$ denotes the spatial resolution of a 2D grid and each pixel encodes the attributes of a single Gaussian primitive and g denotes the number of Gaussian parameters. Here, V_{in} represents the number of grids, each corresponding to a camera

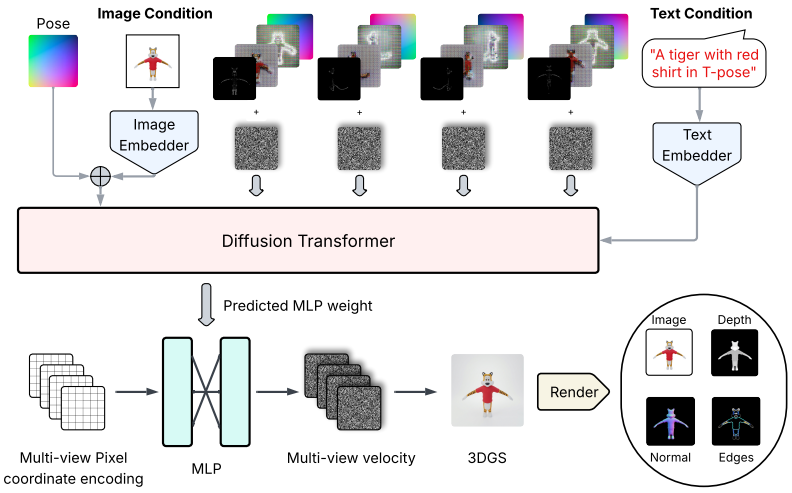


Fig. 1: Pipeline Overview. PixGS directly denoises 3D Gaussian attribute tensors conditioned on image and text prompts utilizing 2D priors from Pixel Diffusion models. $\oplus$ denotes the concatenation of features.

viewpoint that spatially covers the object, resulting in a total of $V_{in} \times H \times W$ Gaussians. Intuitively, this representation is analogous to a multi-view image set where Gaussian attributes replace standard RGB channels, thereby facilitating the seamless adaptation of 2D pixel-space diffusion models to 3D generation. To bootstrap the training process, we obtain pseudo-labels from off-the-shelf 3DGS reconstructors that follow a Splatter Image-style [39] representation, mapping multi-view images into our target attribute space. Specifically, we utilize the reconstruction capability of GSRecon [21], fine-tuning it to generate pseudo-labels for training our generative model.

4.2 Generative model

We adapt PixNerd [44] architecture to the 3D domain, leveraging its extensive pre-trained priors to facilitate Gaussian generation. By building upon this framework, our model inherits rich 2D knowledge acquired from a large-scale corpus of approximately 45 million images, which provides a robust foundation for capturing complex visual details.

Model Architecture An overview of our proposed PixGS pipeline is illustrated in Fig. 1. Given a batch size B of Gaussian attribute tensors $\mathcal{G}_1 \in \mathbb{R}^{B \times V_{in} \times g \times H \times W}$, we define a linear probability path following Flow Matching formulation [22]. The forward diffusion process interpolates between the data

tensor $\mathcal{G}_1$ and a Gaussian noise tensor $\mathcal{G}_0 \sim \mathcal{N}(0, \mathbf{I})$ as follows:

$$\mathcal{G}_t = t\mathcal{G}_1 + (1-t)\mathcal{G}_0, \quad t \in [0,1]. \tag{5}$$

To provide the model with geometric guidance, we augment the noisy tensor with Plücker coordinates [37] $\mathcal{P} \in \mathbb{R}^{V_{in} \times d_{pl} \times H \times W}$ representing the camera rays for each viewpoint. The resulting augmented tensor is denoted as $\mathcal{G}'_t \in \mathbb{R}^{B \times V_{in} \times c \times H \times W}$, where $c = g + d_{pl}$ is the total channel dimension.

Following the pixel-space diffusion convention, we partition the spatial dimensions of each viewpoint into large non-overlapping patches of size $P \times P$ (e.g. $P = 16$). These patches are flattened and concatenated to form a sequence of tokens $\mathbf{X} \in \mathbb{R}^{(B \cdot V_{in}) \times L \times D}$, where $L = \frac{HW}{P^2}$ is the sequence length per view and $D = P^2 \cdot c$ is the token dimensionality.

To ensure cross-view geometric consistency, we extend the standard Attention block to a Multi-view Attention block [13]. By reshaping the token sequence to $\mathbf{X} \in \mathbb{R}^{B \times (V_{in} \cdot L) \times D}$ before attention computation, each patch is allowed to attend to all other patches across all viewpoints of the same 3D object. We also adapt RoPE2d [38] in Pixel Diffusion into a Multi-view RoPE2d. Rather than applying embeddings independently to each view, we treat V_{in} viewpoints as spatially aligned segments of a high-resolution composite image. Specifically, we scale the RoPE2d coordinate manifold to a virtual resolution of $(\frac{V_{in}}{2}H) \times (\frac{V_{in}}{2}W)$ to accommodate the collective resolution of all views, assigning unique positional embeddings across $V_{in} \cdot L$ tokens. This combination allows the transformer to learn relative spatial relationships not only within a single view but across the entire multi-view collective.

Following the transformer blocks, the final hidden states $\mathbf{X}_v^n$, representing the n-th patch of viewpoint v-th, are used to parameterize a local neural field. We predict the weights $\mathcal{W}_v^n$ of a patch-specific MLP via a linear projection of the conditioned hidden states:

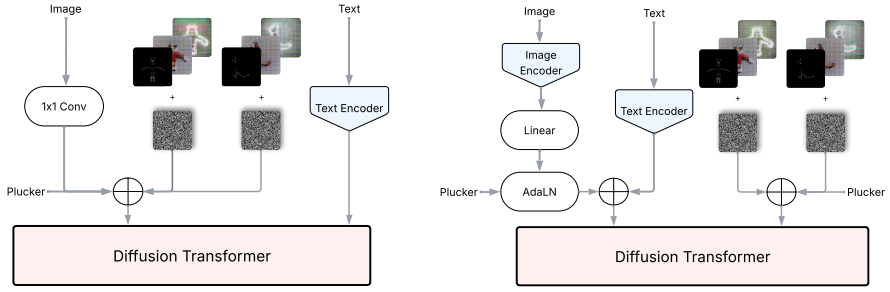
$$\mathcal{W}_v^n = \text{Linear}(\text{SiLU}(\mathbf{X}_v^n)). \tag{6}$$

To denoise Gaussian splats directly in pixel space, the velocity $\mathbf{v}$ at a specific multi-view pixel coordinate (v, i, j) within the patch is decoded as:

$$\mathbf{v}^n(v,i,j) = \text{MLP}\left(\text{Concat}[\text{PE}(v,i,j), \mathcal{G}'^n_t(v,i,j)] \mid \mathcal{W}_v^n\right), \tag{7}$$

where PE denotes a coordinate encoding and $\mathcal{G}'^n_t$ is the noisy input at patch n.

Image-conditioned Paradigms Our objective is to develop a 3D generative framework that can be conditioned on either text or images. However, since text-to-image pixel diffusion models do not natively support visual prompts, we must adapt the architecture to accommodate the image modality. To this end, we explore two widely adopted approaches for image-conditioned generation: View Concatenation [2,21] and Image Prompt Adapter [48,52]. We provide a schematic overview of these approaches in Fig. 2 and a comparison of their performance in our ablation study (Sec. 6.2).



(a) Viewpoint Concatenation **(b)** Image Prompt Adapter

Fig. 2: Paradigms for image-conditioned generation. We explore two strategies: (a) The reference image is integrated as an additional viewpoint (b) Semantic features are extracted and injected via cross-attention. $\oplus$ denotes the concatenation of features.

Viewpoint Concatenation: In this configuration, we incorporate the reference image I_{cond} as a guided viewpoint by projecting its RGB channels to match the Gaussian attribute dimension g and append the Plücker coordinates representing the reference camera pose. This conditioned feature map is then concatenated along the view dimension with the noisy attribute tensor $\mathcal{G}_t$.

Image Prompt Adapter: This approach leverages DINO-v2 [30] to extract features from I_{cond}. We employ an AdaLN [32] module to inject viewpoint information, where embedding is scaled and shifted based on the reference Plücker coordinates. The image-embedding is subsequently projected to match the text-embedding dimensionality and are prepended to the textual prompt tokens.

4.3 Training Objectives

Our model is trained using a multi-task objective that combines generative modeling with pixel-aligned rendering supervision and structural regularization.

Flow Matching Loss: Following Flow Matching [22] formulation, the model v_θ is trained to predict the constant velocity $v_t = \mathcal{G}_1 - \mathcal{G}_0$ that moves noise toward the data manifold:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t,\mathcal{G}_0,\mathcal{G}_1}\left[\|v_\theta(\mathcal{G}_t, t) - (\mathcal{G}_1 - \mathcal{G}_0)\|_2^2\right], \tag{8}$$

where t is uniformly sampled from $[0, 1]$. As analyzed in Sec. 6.1, relying exclusively on flow matching supervision results in poor 3D consistency, as the model fits the pseudo-label distribution without understanding underlying geometric constraints. To ensure physical plausibility, we complement the flow matching objective with direct supervision of 3D appearance and geometry.

Appearance Loss: Although our model produces a large number of Gaussians, many initially appear as low-opacity background artifacts (Fig. 6). To maximize primitive utilization, we perform rendering at twice the spatial resolution of the attribute tensor $\mathcal{G}_1$ ($2H \times 2W$). This super-resolution supervision provides a dense gradient signal, forcing the model to concentrate Gaussian density on valid object surfaces and recover high-frequency details that exceed the base grid resolution.

$$\mathcal{L}_{\text{app}} = \frac{1}{V}\sum_{v=1}^{V}\left(\mathcal{L}_{\text{MSE}}(I_v, I_v^{\text{GT}}) + \lambda_p \mathcal{L}_{\text{LPIPS}}(I_v, I_v^{\text{GT}}) + \mathcal{L}_{\text{MSE}}(M_v, M_v^{\text{GT}})\right), \quad (9)$$

where I_v and M_v are the RGB images and silhouette masks differentiably rendered from the predicted 3D Gaussian tensor $\mathcal{G}_1$ via 3DGS rasterization [56], and V denotes the number of rendered supervision views.

Geometry Loss: We regularize the underlying geometry using surface normal and depth supervision:

$$\mathcal{L}_{\text{geo}} = \frac{1}{V}\sum_{v=1}^{V}\left(\lambda_d \|D_v - D_v^{GT}\|_2 + \lambda_n(1 - \cos(N_v, N_v^{GT}))\right), \quad (10)$$

where D_v and N_v are the rendered depth and surface normal maps, respectively.

High-frequency Loss: To capture actively intricate details, we introduce a Multi-Scale Laplacian of Gaussian (LoG) loss $\mathcal{L}_{\text{LoG}}$. More detailed analysis on $\mathcal{L}_{\text{LoG}}$ will be provided in the Appendix. The LoG loss operates on the rendered images to emphasize high-frequency feature alignment:

$$\mathcal{L}_{\text{LoG}} = \frac{1}{V}\sum_{v=1}^{V}\left(\sum_{\sigma\in\mathcal{S}} \omega_\sigma \|\Delta(G_\sigma * I_v) - \Delta(G_\sigma * I_v^{GT})\|_1\right), \quad (11)$$

where $\mathcal{S}$ denotes the set of standard deviations for the Gaussian kernels, Δ is the Laplace operator and G_σ is a Gaussian kernel, and scale-specific weights $\omega_\sigma \propto \sigma^2$, balances the contribution of sharp features against rendering noise. The total objective is defined as:

$$\mathcal{L}_{\text{total}} = \lambda_f \mathcal{L}_{\text{FM}} + \lambda_a \mathcal{L}_{\text{app}} + \lambda_g \mathcal{L}_{\text{geo}} + \lambda_l \mathcal{L}_{\text{LoG}}. \quad (12)$$

4.4 Training Scheme

We utilize GSRecon solely for early guidance through a three-phase training schedule: (1) flow matching on pseudo-labels, (2) flow matching combined with ground-truth (GT) rendering supervision defined in Sec. 4.3, and (3) GT-only supervision. Under this schedule, pseudo-labels are mainly used to bootstrap convergence and improve training stability, rather than to determine PixGS's final quality. This is because they are limited to the first 50K iterations, whereas

the longest stage, phase 3, relies entirely on GT-only supervision for 150K iterations. Phase 2 then serves as a smooth transition from pseudo-label pretraining to GT-only fine-tuning, improving robustness with limited overhead.

Because 3DGS [16, 17] imposes strict requirements on attribute value ranges and properties: coordinates, color, opacity, and scale must remain within strict range, while rotations must maintain unit norm. Applying rendering losses prematurely when there is significant noise in early iterations causes numerical instability, leading to NaN gradients. Consequently, we use the flow matching loss with guidance from pseudo-labels in early stages to gradually map the model output to 3DGS attribute distribution, facilitating a stable transition to Phase 2 for appearance and geometry supervision. At this stage, because background Gaussians have been suppressed to near-zero opacity, the rendering process is significantly faster, leading to shorter iteration times.

However, the diffusion objective alone cannot perfectly learn the valid 3DGS attribute format. While we L_2-normalize rotations and clamp other attributes during the forward pass, these operations can introduce discrepancies that make rendering losses unreliable. We observe that although diffusion training eventually produces values close to the correct range, the unit-norm rotation property is particularly difficult to learn solely through flow matching. To address this, we introduce an explicit regularization term for the rotation attribute norm:

$$\mathcal{L}_{\text{rot}} = \mathbb{E}\left[(\|\mathbf{r}\|_2 - 1)^2\right], \tag{13}$$

where $\mathbf{r}$ denotes the predicted rotation quaternion. This training scheme ensures numerical stability and facilitates the rapid convergence of our generative model.

5 Experiments

5.1 Experimental Settings

Datasets: We train PixGS on G-Objaverse and G-Objaverse-XL Alignment [34], which are curated high-quality subsets derived from Objaverse [8] and Objaverse-XL [7], respectively. We further refined these collections by filtering assets based on aesthetic scores, resulting in a final training corpus of over 500K 3D objects. Each asset is rendered from 38 diverse viewpoints to ensure comprehensive spatial coverage, with corresponding textual descriptions provided by Cap3D [24].

Text-Conditioned: To evaluate the quality of our text-to-3D synthesis, we utilize 300 prompts from T3Bench [11] benchmark. These prompts are categorized into three levels of complexity: single object, objects with surroundings, and multi objects. We quantify the alignment between the input text and the rendered 3D results using CLIP similarity [35] and CLIP R-Precision [31] (ViT-B/32). Furthermore, we employ ImageReward [50] to assess the generated assets according to human aesthetic preferences.

Fig. 3: Qualitative Results and Comparisons on Text-conditioned 3D Generation. Best viewed **ZOOMED-IN**.

Image-conditioned: For tasks involving image-conditioned generation and reconstruction, we benchmark our model on 300 randomly selected objects from GSO [9] dataset, which remain unseen during training. We evaluate the quality of the synthesized views against GT renders using standard reconstruction metrics, including PSNR, SSIM, and LPIPS [59], all quantitative results are computed and averaged across multiple sampled viewpoints.

5.2 Text-conditioned Generation

Baseline: We benchmark our method against 5 state-of-the-art baselines capable of generating 3DGS representations. DiffSplat [21] adapts pre-trained text-to-image latent diffusion models to generate Structured Splats. Among 3D-native models, TRELLIS [48] generates 3DGS from a sparse voxel latent space, while GaussianCube [57] employs a 3D UNet to learn Gaussian distributions within a predefined voxel grid. We also compare against DreamGaussian [41], an optimization-based framework that utilizes progressive densification, and LGM [40], a feed-forward reconstruction model that we adapt for text-to-3D tasks by pairing it with a multi-view diffusion model [36].

Results and Comparisions: As demonstrated in Tab. 1 and Fig. 3, PixGS exhibits strong prompt alignment and visual quality among text-conditioned 3DGS generators. By inheriting 2D generative priors and bypassing latent compression such in DiffSplat, our approach successfully handles complex prompts. Conversely, 3D-native architectures are hindered by the limited availability of text-3D pairs for training from scratch. Both reconstruction and optimization-based frameworks suffer from multi-view inconsistencies due to their reliance on a 2D diffusion models, which struggle to maintain geometric coherence.

Table 1: Quantitative evaluations on T3Bench. † denotes reconstruction-based methods requiring additional text-conditioned multi-view generative models. GC and DG denote GaussianCube and DreamGaussian, respectively. **Bold** indicates best, underline second best.

Method	DiffSplat [21]		TRELLIS [48]		GC	LGM†	DG	PixGS
	SD-1.5	SD-3.5	L	XL	[57]	[40]	[41]	(Ours)
Single Object								
↑CLIP Sim%	30.63	30.99	28.77	28.75	27.35	29.96	24.78	**31.98**
↑CLIP R-Pre%	78.50	84.75	68.50	69.50	50.75	78.00	37.75	**88.75**
↑ImgReward	-0.490	-0.196	-1.033	-0.961	-1.521	-0.720	-1.635	**-0.171**
Single w/ Surr								
↑CLIP Sim%	30.20	30.91	28.37	28.35	25.60	27.79	23.81	**31.32**
↑CLIP R-Pre%	86.00	**87.75**	73.50	70.00	47.25	55.00	41.50	87.25
↑ImgReward	-1.063	-0.447	-1.604	-1.518	-2.063	-1.772	-1.953	**-0.341**
Multiple Objects								
↑CLIP Sim%	27.62	29.44	26.28	26.58	23.93	27.07	23.97	**30.45**
↑CLIP R-Pre%	69.25	74.00	48.00	49.75	26.25	51.00	28.50	**77.25**
↑ImgReward	-1.585	-0.805	-1.906	-1.883	-2.193	-1.731	-2.069	**-0.740**
↓Latency (s)	**1**	3	5	7	3	6	47	**1**

5.3 Image-conditioned Generation

Baseline: For the image-conditioned task, we compare our method against four representative baselines: DiffSplat [21], TRELLIS [48], a single-stage pipeline DiffusionGS [2], and LGM [40] as a reconstruction-based method that relies on a separate multi-view diffusion model [36] conditioned on an image prompt.

Results and Comparisions: Quantitative and qualitative results (Tab. 2, Fig. 4) show that PixGS obtains favorable results in alignment and geometric fidelity. Unlike DiffusionGS, which struggles to generalize to unseen images due to being trained from scratch on limited corpus of synthetic 3D views. Furthermore, we observe that 3D-native models like TRELLIS face challenges in reference-view alignment. Because of operating in a canonical latent space without explicit viewpoint-conditional supervision, it often fails to reproduce the exact spatial orientation and perspective required to match the ground-truth reference image.

6 Ablation

We analyze the efficacy of our image-conditioning strategies, loss functions, multi-phase training schedule and pseudo-label dependence through a series of ablation experiments. These models are trained on the G-Objaverse [8, 34] dataset (265K assets) and evaluated on subset of 1,000 samples from G-Objaverse-XL Alignment [7, 34].

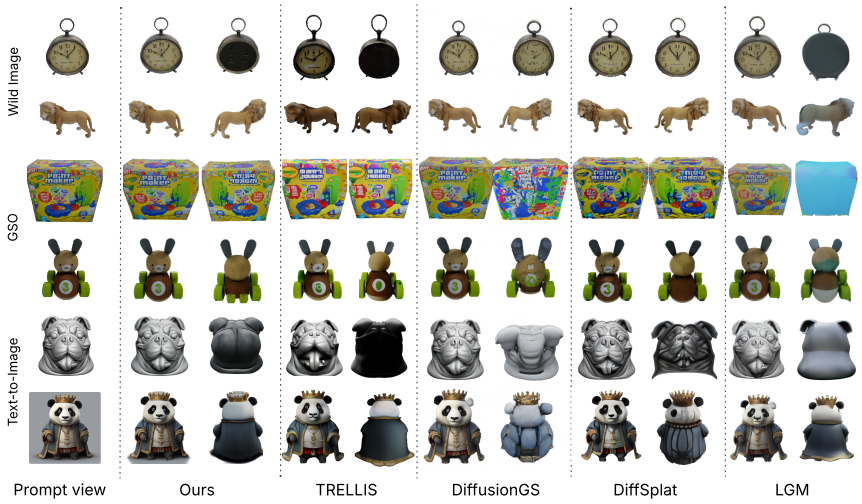


Fig. 4: Qualitative Results and Comparisons on Image-conditioned 3D Generation. Best viewed **ZOOMED-IN**.

Table 2: Quantitative evaluations on the GSO dataset. † denotes methods requiring additional image-conditioned multi-view generative models. **Bold** indicates best result, underline second best.

Method	DiffSplat [21] SD-1.5	DiffSplat [21] SD-3.5	DiffusionGS [2]	TRELLIS-L [48]	LGM† [40]	PixGS (Ours)
↑PSNR	19.57	19.59	16.53	17.70	15.61	**21.21**
↑SSIM	0.810	0.811	0.770	0.797	0.764	**0.842**
↓LPIPS	0.159	0.158	0.217	0.172	0.245	**0.123**
↓Latency (s)	**1**	3	12	6	2	**1**

6.1 Training Strategy

Tab. 4 highlights that adapting 2D pixel-space diffusion to the 3DGS modality requires more than naive distribution matching. Although optimizing only the flow-matching objective on pseudo-labels leads to rapid loss convergence, it does not guarantee high-quality 3D synthesis. Since our model operates directly at the splat level, small errors in attribute prediction, especially for near-zero opacity values, can produce persistent "floater" artifacts (see Fig. 6). Thus, pseudo-label supervision is most useful as early guidance, while the main training signal comes from ground-truth appearance supervision. Geometry loss is auxiliary refinement for depth and normal quality, and the rotation regularizer stabilizes Gaussian parameterization, preventing numerical instability. Furthermore, the integration of the Multi-Scale LoG loss encourages the model to preserve high-

Table 3: Ablation of image conditioning paradigms. We compare Viewpoint Concatenation and Image Prompt Adaptation. **Bold** indicates best result, underline second best.

Paradigm	↑PSNR	↑SSIM	↓LPIPS	↓Params (B)
View Concatenation	**30.36**	0.969	**0.022**	**1.2**
IP-Adapter	30.31	**0.977**	0.023	1.4

frequency details, as shown in Fig. 5. Complementary to this, Tab. 5 shows that the three-phase training schedule gradually shifts the model from pseudo-label-guided pretraining to ground-truth-only supervision, thereby decoupling final generation quality from pseudo-label constraints.

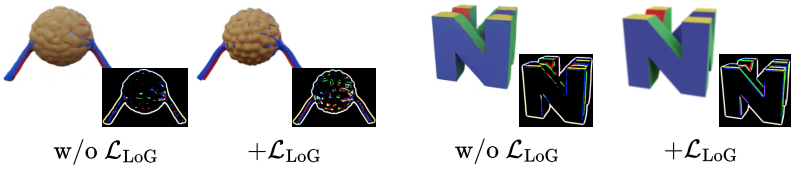


Fig. 5: Effect of $\mathcal{L}_{\text{LoG}}$. $\mathcal{L}_{\text{LoG}}$ promotes the recovery of high-frequency details and mitigates over-smoothing, resulting in sharper geometric boundaries and texture.

6.2 Image-conditioned Paradigms

We systematically compare the two image-conditioning strategies in Tab. 3. While both paradigms yield comparable performance, Viewpoint Concatenation is more parameter-efficient, facilitating training with larger batch sizes. We also find that Viewpoint Concatenation achieves higher PSNR, suggesting higher pixel-level fidelity. This is likely due to the preservation of the reference image's 2D spatial structure through view-axis concatenation, which enables denser and more direct cross-view attention. Conversely, Image Prompt Adapter approach demonstrates a slight advantage in capturing global structural patterns by leveraging the powerful priors of an image encoder.

6.3 Pseudo-label Dependence

To assess PixGS's dependence on pseudo-labels from GSRecon [21], we replace them with lower-quality LGM [40] pseudo-labels. As shown in Tab. 6, PixGS trained with LGM pseudo-labels achieves similar GSO performance, while requiring only about 10% more training to reach plausible results. This suggests that our model's final quality is not strongly tied to the capability of the reconstructor. Therefore, we prefer GSRecon because it is compact and efficient, not

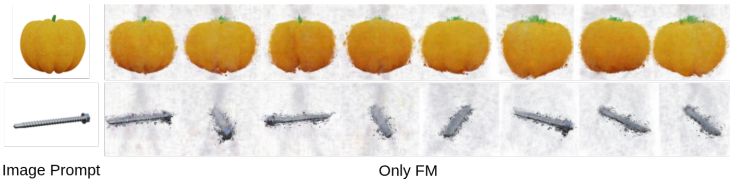


Fig. 6: Limitations of standalone Diffusion Loss supervision. Despite the low loss values in Phase 1, the resulting 3DGS suffers from significant “floater" artifacts.

Table 4: Ablation of supervision losses and regularization. **Bold** indicates best result.

	↑PSNR	↑SSIM	↓LPIPS
Only $\mathcal{L}_{FM}$	16.04	0.708	0.496
+ $\mathcal{L}_{app}$	NaN	NaN	NaN
+ $\mathcal{L}_{rot}$	30.11	**0.969**	0.027
+ $\mathcal{L}_{geo}$	<u>30.24</u>	<u>0.967</u>	<u>0.025</u>
+ $\mathcal{L}_{LoG}$	**30.36**	**0.969**	**0.022**

Table 5: Effectiveness of the three-phase training curriculum. **Bold** indicates best result.

	↑PSNR	↑SSIM	↓LPIPS
Phase 1	16.04	0.708	0.496
Phase 2	<u>30.36</u>	<u>0.969</u>	<u>0.022</u>
Phase 3	**31.15**	**0.987**	**0.019**

Table 6: Effect of pseudo-label source on PixGS's final quality on GSO. **Bold** indicates best result.

Pseudo-label	↑PSNR	↑SSIM	↓LPIPS	↓Params (M)
LGM [40]	21.15	**0.844**	0.125	415
GSRecon [21]	**21.21**	0.842	**0.123**	**42**

because it sets the quality upper bound. Moreover, using pseudo-labels as early guidance avoids lengthy rendering-only optimization from noisy 3DGS point clouds with invalid parameters. Since higher-quality pseudo-labels can accelerate convergence, we incur a small, controlled GSRecon fine-tuning cost to improve training stability. This cost remains substantially lower than training a VAE required by existing latent-based methods.

7 Conclusion

We have presented PixGS, a single-stage model for 3DGS generation that bypasses the limitations of latent-space pipelines. By leveraging 2D pixel-space diffusion priors and a tailored suite of loss functions, including the multi-scale Laplacian of Gaussian loss, PixGS produces 3D objects with detailed appearance and strong geometric fidelity. Our three-phase training schedule and rotation regularization ensure stable convergence, enabling efficient generation of high-quality assets from complex text and image prompts in approximately one second. Overall, PixGS highlights the potential of direct splat-level denoising as a fast and effective approach for 3D generative modeling.

References

1. Ahmed, N., Natarajan, T., Rao, K.: Discrete cosine transform. IEEE Transactions on Computers **C-23**(1), 90–93 (1974). https://doi.org/10.1109/T-C.1974.223784
2. Cai, Y., Zhang, H., Zhang, K., Liang, Y., Ren, M., Luan, F., Liu, Q., Kim, S.Y., Zhang, J., Zhang, Z., Zhou, Y., Zhang, Y., Yang, X., Lin, Z., Yuille, A.: Baking gaussian splatting into diffusion denoiser for fast and scalable single-stage image-to-3d generation and reconstruction (2025), https://arxiv.org/abs/2411.14384
3. Charatan, D., Li, S., Tagliasacchi, A., Sitzmann, V.: pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction (2024), https://arxiv.org/abs/2312.12337
4. Chen, A., Xu, H., Esposito, S., Tang, S., Geiger, A.: Lara: Efficient large-baseline radiance fields (2024), https://arxiv.org/abs/2407.04699
5. Chen, S., Ge, C., Zhang, S., Sun, P., Luo, P.: Pixelflow: Pixel-space generative models with flow (2025), https://arxiv.org/abs/2504.07963
6. Chen, Z., Wang, F., Wang, Y., Liu, H.: Text-to-3d using gaussian splatting (2024), https://arxiv.org/abs/2309.16585
7. Deitke, M., Liu, R., Wallingford, M., Ngo, H., Michel, O., Kusupati, A., Fan, A., Laforte, C., Voleti, V., Gadre, S.Y., VanderBilt, E., Kembhavi, A., Vondrick, C., Gkioxari, G., Ehsani, K., Schmidt, L., Farhadi, A.: Objaverse-xl: A universe of 10m+ 3d objects (2023), https://arxiv.org/abs/2307.05663
8. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., VanderBilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3d objects (2022), https://arxiv.org/abs/2212.08051
9. Downs, L., Francis, A., Koenig, N., Kinman, B., Hickman, R., Reymann, K., McHugh, T.B., Vanhoucke, V.: Google scanned objects: A high-quality dataset of 3d scanned household items (2022), https://arxiv.org/abs/2204.11918
10. Fu, X., Yin, W., Hu, M., Wang, K., Ma, Y., Tan, P., Shen, S., Lin, D., Long, X.: Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image (2024), https://arxiv.org/abs/2403.12013
11. He, Y., Bai, Y., Lin, M., Zhao, W., Hu, Y., Sheng, J., Yi, R., Li, J., Liu, Y.J.: T^3bench: Benchmarking current progress in text-to-3d generation (2024), https://arxiv.org/abs/2310.02977
12. Hong, F., Tang, J., Cao, Z., Shi, M., Wu, T., Chen, Z., Yang, S., Wang, T., Pan, L., Lin, D., Liu, Z.: 3dtopia: Large text-to-3d generation model with hybrid diffusion priors (2024), https://arxiv.org/abs/2403.02234
13. Huang, Z., Guo, Y.C., Wang, H., Yi, R., Ma, L., Cao, Y.P., Sheng, L.: Mv-adapter: Multi-view consistent image generation made easy (2024), https://arxiv.org/abs/2412.03632
14. Jun, H., Nichol, A.: Shap-e: Generating conditional 3d implicit functions (2023), https://arxiv.org/abs/2305.02463
15. Ke, B., Obukhov, A., Huang, S., Metzger, N., Daudt, R.C., Schindler, K.: Repurposing diffusion-based image generators for monocular depth estimation (2024), https://arxiv.org/abs/2312.02145
16. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering (2023), https://arxiv.org/abs/2308.04079
17. Kheradmand, S., Rebain, D., Sharma, G., Sun, W., Tseng, J., Isack, H., Kar, A., Tagliasacchi, A., Yi, K.M.: 3d gaussian splatting as markov chain monte carlo (2025), https://arxiv.org/abs/2404.09591

18. Lan, Y., Hong, F., Zhou, S., Yang, S., Meng, X., Chen, Y., Lyu, Z., Dai, B., Pan, X., Loy, C.C.: Ln3diff++: Scalable latent neural fields diffusion for speedy 3d generation (2025). https://doi.org/https://doi.org/10.1109/TPAMI.2025.3633073, https://arxiv.org/abs/2403.12019
19. Li, W., Chen, R., Chen, X., Tan, P.: Sweetdreamer: Aligning geometric priors in 2d diffusion for consistent text-to-3d (2023), https://arxiv.org/abs/2310.02596
20. Lin, C.H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.Y., Lin, T.Y.: Magic3d: High-resolution text-to-3d content creation (2023), https://arxiv.org/abs/2211.10440
21. Lin, C., Pan, P., Yang, B., Li, Z., Mu, Y.: Diffsplat: Repurposing image diffusion models for scalable gaussian splat generation (2025), https://arxiv.org/abs/2501.16764
22. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling (2023), https://arxiv.org/abs/2210.02747
23. Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: Syncdreamer: Generating multiview-consistent images from a single-view image (2024), https://arxiv.org/abs/2309.03453
24. Luo, T., Rockwell, C., Lee, H., Johnson, J.: Scalable 3d captioning with pretrained models (2023), https://arxiv.org/abs/2306.07279
25. Ma, Z., Wei, L., Wang, S., Zhang, S., Tian, Q.: Deco: Frequency-decoupled pixel diffusion for end-to-end image generation (2025), https://arxiv.org/abs/2511.19365
26. Ma, Z., Xu, R., Zhang, S.: Pixelgen: Pixel diffusion beats latent diffusion with perceptual loss (2026), https://arxiv.org/abs/2602.02493
27. Marr, D., Hildreth, E.: Theory of edge detection. Proceedings of the Royal Society of London. B. Biological Sciences **207**(1167), 187–217 (02 1980). https://doi.org/10.1098/rspb.1980.0020, https://doi.org/10.1098/rspb.1980.0020
28. Meng, X., Wang, C., Lei, J., Daniilidis, K., Gu, J., Liu, L.: Zero-1-to-g: Taming pretrained 2d diffusion model for direct 3d generation (2025), https://arxiv.org/abs/2501.05427
29. Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., Chen, M.: Point-e: A system for generating 3d point clouds from complex prompts (2022), https://arxiv.org/abs/2212.08751
30. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: Dinov2: Learning robust visual features without supervision (2024), https://arxiv.org/abs/2304.07193
31. Park, D.H., Azadi, S., Liu, X., Darrell, T., Rohrbach, A.: Benchmark for compositional text-to-image synthesis. In: Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track (2021)
32. Peebles, W., Xie, S.: Scalable diffusion models with transformers (2023), https://arxiv.org/abs/2212.09748
33. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion (2022), https://arxiv.org/abs/2209.14988
34. Qiu, L., Chen, G., Gu, X., Zuo, Q., Xu, M., Wu, Y., Yuan, W., Dong, Z., Bo, L., Han, X.: Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d (2023), https://arxiv.org/abs/2311.16918

35. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021), https://arxiv.org/abs/2103.00020
36. Shi, Y., Wang, P., Ye, J., Long, M., Li, K., Yang, X.: Mvdream: Multi-view diffusion for 3d generation (2024), https://arxiv.org/abs/2308.16512
37. Sitzmann, V., Rezchikov, S., Freeman, W.T., Tenenbaum, J.B., Durand, F.: Light field networks: Neural scene representations with single-evaluation rendering (2022), https://arxiv.org/abs/2106.02634
38. Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding (2023), https://arxiv.org/abs/2104.09864
39. Szymanowicz, S., Rupprecht, C., Vedaldi, A.: Splatter image: Ultra-fast single-view 3d reconstruction (2024), https://arxiv.org/abs/2312.13150
40. Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., Liu, Z.: Lgm: Large multi-view gaussian model for high-resolution 3d content creation (2024), https://arxiv.org/abs/2402.05054
41. Tang, J., Ren, J., Zhou, H., Liu, Z., Zeng, G.: Dreamgaussian: Generative gaussian splatting for efficient 3d content creation (2024), https://arxiv.org/abs/2309.16653
42. Tang, S., Zhang, F., Chen, J., Wang, P., Furukawa, Y.: Mvdiffusion: Enabling holistic multi-view image generation with correspondence-aware diffusion (2023), https://arxiv.org/abs/2307.01097
43. Team, T.H.: Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details (2025), https://arxiv.org/abs/2506.16504
44. Wang, S., Gao, Z., Zhu, C., Huang, W., Wang, L.: Pixnerd: Pixel neural field diffusion (2025), https://arxiv.org/abs/2507.23268
45. Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., Zhu, J.: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation (2023), https://arxiv.org/abs/2305.16213
46. Wang, Z., Wang, Y., Chen, Y., Xiang, C., Chen, S., Yu, D., Li, C., Su, H., Zhu, J.: Crm: Single image to 3d textured mesh with convolutional reconstruction model (2024), https://arxiv.org/abs/2403.05034
47. Xiang, J., Chen, X., Xu, S., Wang, R., Lv, Z., Deng, Y., Zhu, H., Dong, Y., Zhao, H., Yuan, N.J., Yang, J.: Native and compact structured latents for 3d generation (2025), https://arxiv.org/abs/2512.14692
48. Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., Yang, J.: Structured 3d latents for scalable and versatile 3d generation (2025), https://arxiv.org/abs/2412.01506
49. Xu, J., Cheng, W., Gao, Y., Wang, X., Gao, S., Shan, Y.: Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models (2024), https://arxiv.org/abs/2404.07191
50. Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: Learning and evaluating human preferences for text-to-image generation (2023), https://arxiv.org/abs/2304.05977
51. Xu, Y., Shi, Z., Yifan, W., Chen, H., Yang, C., Peng, S., Shen, Y., Wetzstein, G.: Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation (2024), https://arxiv.org/abs/2403.14621
52. Ye, H., Zhang, J., Liu, S., Han, X., Yang, W.: Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models (2023), https://arxiv.org/abs/2308.06721

53. Yi, T., Fang, J., Wang, J., Wu, G., Xie, L., Zhang, X., Liu, W., Tian, Q., Wang, X.: Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models (2024), https://arxiv.org/abs/2310.08529
54. Yu, M., Lu, T., Xu, L., Jiang, L., Xiangli, Y., Dai, B.: Gsdf: 3dgs meets sdf for improved rendering and reconstruction (2024), https://arxiv.org/abs/2403.16964
55. Yu, Y., Xiong, W., Nie, W., Sheng, Y., Liu, S., Luo, J.: Pixeldit: Pixel diffusion transformers for image generation (2026), https://arxiv.org/abs/2511.20645
56. Zhang, B., Fang, C., Shrestha, R., Liang, Y., Long, X., Tan, P.: Rade-gs: Rasterizing depth in gaussian splatting (2024), https://arxiv.org/abs/2406.01467
57. Zhang, B., Cheng, Y., Yang, J., Wang, C., Zhao, F., Tang, Y., Chen, D., Guo, B.: Gaussiancube: A structured and explicit radiance representation for 3d generative modeling (2024), https://arxiv.org/abs/2403.19655
58. Zhang, L., Wang, Z., Zhang, Q., Qiu, Q., Pang, A., Jiang, H., Yang, W., Xu, L., Yu, J.: Clay: A controllable large-scale generative model for creating high-quality 3d assets (2024), https://arxiv.org/abs/2406.13897
59. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric (2018), https://arxiv.org/abs/1801.03924
60. Zhao, Z., Liu, W., Chen, X., Zeng, X., Wang, R., Cheng, P., Fu, B., Chen, T., Yu, G., Gao, S.: Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation (2023), https://arxiv.org/abs/2306.17115